Visual Testing with PyCharm and pytest

Brian Okken & Paul Everitt

May 11, 2018

Who we are

Paul Everitt

PyCharm and WebStorm Developer Advocate at JetBrains

Brian Okken

Author of "Python Testing with pytest"
Host of "Test & Code" podcast
Co-Host of "Python Bytes" podcast
Lead Software Engineer at Rohde & Schwarz

TDD, pytest, and PyCharm

a lot of ground to cover

- TDD in 4 slides
- why pytest
- why PyCharm
- demo as much as we can

SCORE 0000   TIME   0:25

ALT   0025   X VELOCITY   2
FUEL   0937   Y VELOCITY   106

(courtesy http://moonlander.seb.ly/)

TDD

Design->Develop->Test

doesn't work

at least ... it's not fun

Test First

~~Design -> Develop -> Test~~

Test -> Design & Develop

TDD - traditional

- Red: Write a failing test
- Green: Make it pass
- Refactor: Clean up your code

TDD according to Brian

- Red: Learn what it's like to use your code
- Green: Learn one way to implement the code
- Refactor: Make code you are proud of

 (and don't forget to refactory your tests, too)

Let's test something

The cards project

```
$ cards add -o okken "Prepare for PyCon talk"
$ cards add -o okken "Order stickers"
$ cards add -o okken "Order business cards"
$ cards add -o okken "pack"
$ cards update 2 --done True
$ cards update 3 --done True
$ cards list
  ID   owner     done     summary
----   -------   ------   ----------------------
   1   okken              Prepare for PyCon talk
   2   okken     x        Order stickers
   3   okken     x        Order business cards
   4   okken              pack
```

for those of you following along at home

```
$ git clone -b pycon_2018 https://github.com/okken/cards.git
$ cd cards
$ python3.6 -m venv venv --prompt cards
$ source venv/bin/activate  # venv\Scripts\activate  (if
Windows)
(cards) $ pip install -r requirements_dev.txt
```

```
cards
├── src
│   └── cards
│       ├── __init__.py
│       ├── cardsdb.py
│       └── cli.py
├── tests
│   ├── api
│   │   ├── conftest.py
│   │   ├── test_alac.py
│   │   ├── test_list_filter.py
│   │   └── test_tracer_bullets.py
│   └── cli
│       ├── __init__.py
│       ├── conftest.py
│       └── test_alac.py
```

pytest in action

```
(cards) $ pytest --cov=src
================== test session starts ==================
plugins: cov-2.5.1
collected 40 items

tests/api/test_alac.py .                              [  2%]
tests/api/test_list_filter.py ...................     [ 50%]
tests/api/test_tracer_bullets.py .......             [ 67%]
tests/cli/test_alac.py .                              [ 70%]
tests/cli/test_list_format.py ......                 [ 85%]
tests/cli/test_tracer_bullets.py ......              [100%]

----------- coverage: -----------
Name                     Stmts    Miss Branch BrPart   Cover
-----------------------------------------------------------
src/cards/__init__.py        2       0      0      0    100%
```

pytest allows you to run a subset of tests

```
$ # one directory
$ pytest tests/cli

$ # one test module
$ pytest tests/cli/test_tracer_bullets.py

$ # one test function
$ pytest tests/cli/test_tracer_bullets.py::test_count
```

Some other ways to run a subset of tests

```
(cards) $ pytest --help
...
  -k EXPRESSION          just tests matching this expression
  -m MARKEXPR            run tests w/ given mark expression
  -x, --exitfirst        exit on first error/failed test
  --maxfail=num          exit after first num failures/errors.
  --lf, --last-failed    rerun failed tests from last run
  --ff, --failed-first   same, but then run the rest
  --nf, --new-first      run new test files, then the rest
```
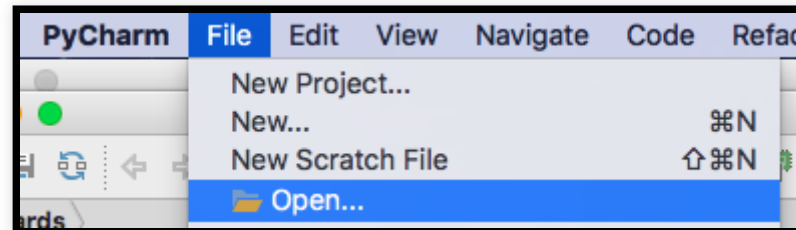
More development speedups

- Fixtures: setup, teardown, resource and test data initialization, …
- Parametrization: many test cases using the same test function
- Hooks and Plugins: the sky is the limit
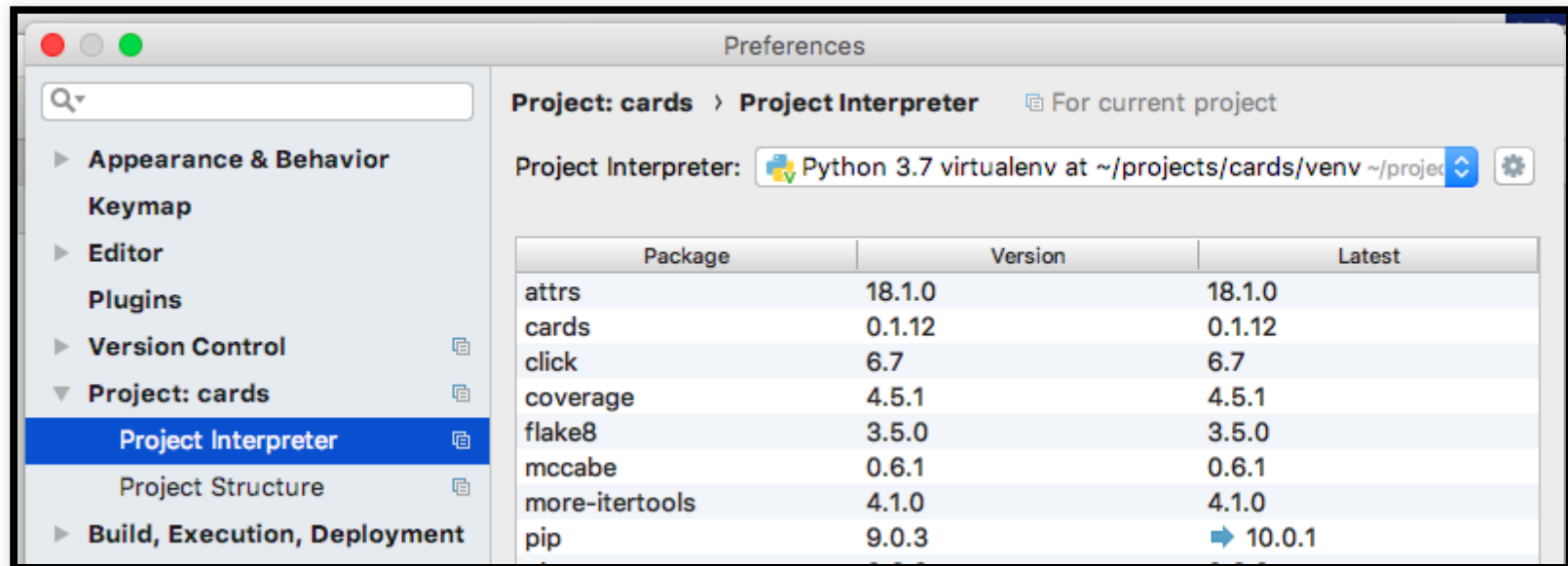
PyCharm

Some Setup

- PyCharm Community edition
- Use File->Open for existing projects
- Check the interpreter setting
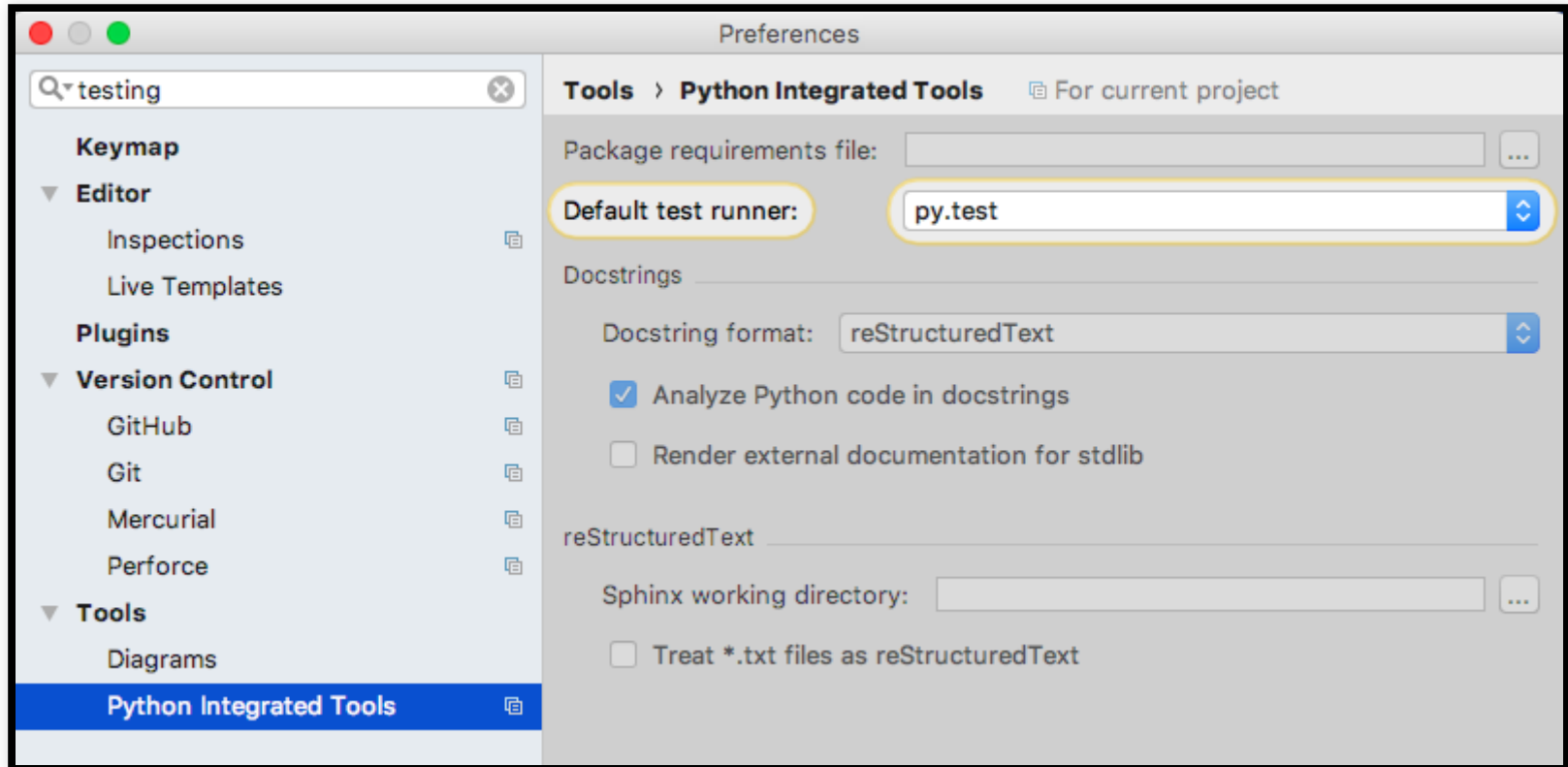- Check the default test runner

Use File->Open for existing projects

Check the interpreter setting

Check the default test runner

Demo

In the demo

- run at different levels
- see test durations
- auto-running tests
- run only failing tests
- jump directly the test failure
- see tests and code at the same time (TDD mode)
- debug tests and code together
- run REPL during a debug session

The END-ish

More info

- PyCharm
  - https://www.jetbrains.com/pycharm/
- Python Testing with pytest
  - https://pragprog.com/book/bopytest/python-testing-with-pytest
- pytest
  - https://docs.pytest.org
  - https://stackoverflow.com/search?q=pytest
- Lessons about testing and TDD from Kent Beck
  - http://testandcode.com/23
- booths
- open space session